Finding Experts Using Wikipedia

Gianluca Demartini

L3S Research Center
Leibniz Universität Hannover
Appelstrasse 9a D-30167 Hannover, Germany
demartini@l3s.de

Abstract. When we want to find experts on the Web we might want to search where the knowledge is created by the users. One of such knowledge repository is Wikipedia. People expertises are described in Wikipedia pages and also the Wikipedia users can be considered experts on the topics they produce content on. In this paper we propose algorithms to find experts in Wikipedia. The two different approaches are finding experts in the Wikipedia content or among the Wikipedia users. We also use semantics from WordNet and Yago in order to disambiguate expertise topics and to improve the retrieval effectiveness. In the end, we show how our methodology can be implemented in a system in order to improve the expert retrieval effectiveness.

1 Introduction

The Web can be viewed as a repository of knowledge produced by experts. Of course, there are several issues on the entire Web such as spam or advertisements which are not present in the common Enterprise scenario. However, there are subparts of Web where these problems are somehow mitigated.

In previous works the problem finding experts in online communities has been already studied [16, 3]. A different domain in which have been sought for experts is the one for the computer science domain based on DBLP [10]. In this paper we propose to search for experts in a domain made of community-maintained artefacts of lasting value (CALV). One example of a dataset made of CALV is Wikipedia[1]. In this community the users proactively produce content using their knowledge about some topics. Other examples of such communities are IMDb[2] and Slashdot[3].

Two approaches are possible to find experts in Wikipedia (see Figure 1). We describe in Section 2 how it is possible to find experts using the Wikipedia content. Looking in the articles describing people we can build an expert profile for each individual that can be later retrieved using a query describing the topic of expertise in which we are looking for experts. The second option is to search for experts among the Wikipedia users (see Section 3). The assumption in this

[1] http://www.wikipedia.org/
[2] http://www.imdb.com/
[3] http://slashdot.org/

case is that the users are somehow experts on the topic they write articles about. In this way, looking also at the edit history of each user, we can build expert profiles for them that can be later retrieved by a regular expert search system (see for example [7, 4, 17, 12, 16]). The solution we propose is similar to [6] where

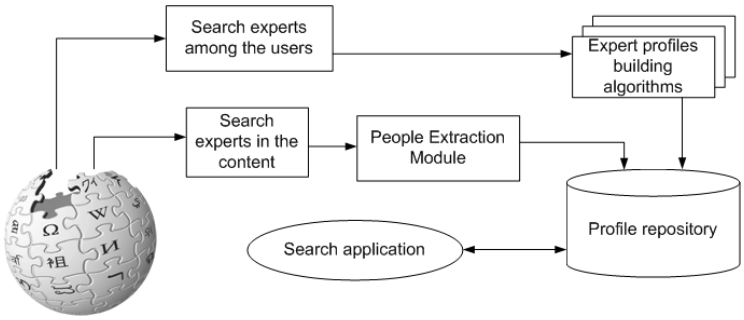

Fig. 1. The two possible approaches to find experts in Wikipedia: search in the content or among the users

the authors propose intelligent task routing algorithms to recommend new tasks to Wikipedia users. We propose to adopt similar user's profiles to model their expertise and to find experts in Wikipedia.

The advantage of this scenario is that we can search for expertise also getting help from a semantic lexical database for the English language such as WordNet[4]. Similarly to what the authors of [14] did, we can use WordNet to disambiguate the topics of expertise (see Section 4). Of course, after proposing the novel algorithms we need to evaluate their performances. In Section 5 we describe the current issues that allow us to perform a standard and sound evaluation. Section 6 concludes the paper summarizing and describing the future steps.

2 Finding Experts in the Content

Wikipedia articles are manually annotated and grouped in categories. One of such category is "Category:People" and all the articles about single individuals are labelled with this category or with its subcategories. In this way we can refer to a person using his related article (if existing).

We propose to create an expert profile for each person in Wikipedia. That is, each article in the category "People" represents a candidate expert. We build for each of these candidates a profile which will describe all her topics of expertise allowing an automatic system to perform a search within this set of profiles. We propose to build this profile as a vector where for each possible topic of expertise there is a score representing the confidence/amount of the candidate expertise.

[4] `http://wordnet.princeton.edu/`

These scores are real numbers in the interval $[0, 1]$ and can be computed using several different evidences. Given a list of topics (see Section 4 on how to refine such a list) we can compute these scores according to the content of the candidate's page. For example we can take as possible expertise topics the words present in the "People" articles and as a score based on the Term Frequency (TF) and Inverse Document Frequency (IDF) values for them.

3 Finding Experts among the Authors

In this paper we focus more on finding experts among Wikipedia users. We describe the appropriate algorithms for this task within the subsequent paragraphs of this section. First, we need to extract expert profiles for the people who participate in the production of the Wikipedia content. Similarly to the system SuggestBot [6], we propose to create for some of the Wikipedia users a profile composed by the title of the articles edited in the past. In the following we describe the possible approaches.

3.1 Naïve Approach

We need algorithms to perform the task of expert finding within the Wikipedia users. One first naïve approach is to use standard Information Retrieval techniques considering the expert profiles (defined as the titles of the edited articles) as standard documents and indexing them using an inverted index. We can then create a query which represents the topic in which we are looking for experts and query the index in order to find a ranked list of people using a TF-IDF similarity measure.

Some more advanced techniques can be used to improve the search effectiveness for example using the link structure as done in the Web for document search. The assumption is that the most linked documents have an higher authority and they are a better representative of the expertise of its authors than other less linked documents. That is, the author of a highly linked/cited article is an expert on the article's topic. In this way we can give different weights to the authors according to the authority of the content they produce (see Algorithm 1). The authority weights of the articles can be computed with standards algorithms which considers the links structure such as PageRank [13] or HITS [11]. One limitation of this approach would appear in the case where the links in Wikipedia do represent *popularity* of topics rather then *authority* of pages as in the Web.

Notation. In the described algorithms we use the following notation:

- $u.score(i)$ represents how good is the item i as representative of the expertise of the user u.
- $u.rsv$ is the score of a user u as returned by an expert search system for a given query. It is used to rank the retrieved experts in the final results.

– $h_{WinN(i)}$ represents the subarticle of the article h composed of a window of text of size N around the link to the article i.

Algorithm 1 Algorithm for defining expert profiles

Require: A set of users U
1. Build expert profiles
for all users u in U **do**
- fetch the list of edited articles
- $u.score(a) = 1$ {concatenate the articles title in one profile P_u}
- store the profile P_u in the repository
for all article a in the profile P_u **do**
- compute/load the *authority weight* of a
$u.score(a) = u.score(a) \cdot authority(a)$ {modify the expertise score of the user u}
end for
end for
2. Index and Search the profiles
return list of ranked profiles

3.2 Using the Citation Network

A more sophisticated algorithm would make further use of the link structure present between the Wikipedia documents. If with the methodology described in Section 3.1 we obtain a small expert profile because the user has participated in the creation of only few articles, we then propose to expand this profile using the citation network with the assumption that the users know something about what they cite. In this way we can define an expert profile with topic weights influenced by the link structure of Wikipedia. In Algorithm 2 we present the strategy which makes use of this information. The algorithm starts considering a profile containing the edited articles. Then, for all the items in the profile, it first considers the ingoing link information. This Algorithm adds to the profile topics extracted from a windows of N (e.g. N=10) words before and after the anchor of the citing documents. For example if the citing document has a part like "A good IDE for java is [Eclipse:Eclipse]." where [target:anchor] indicates a link, we can add *IDE* and *java* to the profile of the authors of the *Eclipse* page. The Algorithm also adds the linked articles with a lower weight into the profile until the profile is big enough.

3.3 Using Users Similarity

A third possibility is to use a technique similar to collaborative filtering. Using a measure of similarity between users (based on co-editing of articles) we can alternatively:

Algorithm 2 Algorithm for expanding expert profiles using the citation network

Require: T is the acceptable number of articles in a user's profile
Require: $size(\cdot)$ function that return the number of articles contained in one profile
Require: run Algorithm 1
  for all profiles P_u **do**
    if $size(P_u) < T$ **then**
      1. Initialize
      for all items i in the user profile P_u **do**
        $u.score(i) = 1$
        for all link from an item h to i **do**
          $u.score(h_{Win10(i)}) = 1$
        end for
      end for
      2. Expand profile
      while $(size(P_u) < T)$ **do**
        for all i with $u.score(i) > 0$ **do**
          for all links from i to an item l **do**
            $u.score(l) = u.score(l) + 1$
          end for
        end for
        - add to P_u all the items l with $u.score(l) > 0$
      end while
    end if
  end for

- extend the expert profile including the topics of expertise of very similar users (see Algorithm 3);
- extend the list of retrieved experts with those similar to the retrieved ones in a pseudo-relevance feedback fashion (see Algorithm 4).

The similarity measure between user's profiles we can use is, for example, the standard Jaccard measure:

$$J(a, b) := \frac{|P_a \cap P_b|}{|P_a \cup P_b|} \tag{1}$$

where P_a, P_b are the sets of articles edited by a user a and a user b respectively.

In Algorithm 3 we first initialize all the items in the user profile to $score = 1$. If needed, we then expand this profile adding the articles edited by the most similar users with a score proportional to the users' similarity.

In Algorithm 4 we first need to run Algorithm 1 in order to have a first list of experts. Then, for each retrieved expert, we search for similar profiles adding them to the results with a Retrieval Status Value (used by the system to rank the experts) proportional to the users' similarity.

4 Using Semantics to Enhance Search

In the scenario we are tackling there are several ways to improve the retrieval effectiveness using different evidences. One of such ways is the use of semantics.

Algorithm 3 Algorithm for expanding expert profiles using the co-editing information

Require: T is the acceptable number of articles in one user's profile
Require: $size(\cdot)$ function that return the number of articles contained in one profile
Require: Z the similarity threshold between two users (e.g. 0.9)
Require: run Algorithm 1
for all profiles P_u **do**
if $size(P_u) < T$ **then**
1. Initialize
for all items i in the user profile P_u **do**
$u.score(i) = 1$
end for
2. Expand profile
while $(size(P_u) < T)$ **do**
for all profiles P_v who have edited any article in common with P_u **do**
$S_{uv} \leftarrow J(u, v)$
if $S_{uv} > Z$ **then**
for all items h in P_v **do**
add h to P_u with $u.score(h) = u.score(h) \cdot S_{uv}$ {weight the authority using the users' similarity}
end for
end if
end for
end while
end if
end for

Algorithm 4 Algorithm for expanding expert search results using relevance feedback

Require: Z the similarity threshold between two users (e.g. 0.9)
1. $retrievedList \leftarrow$ run Algorithm 1
for all profiles P_u in $retrievedList$ **do**
2. Find similar profiles
for all profiles P_v who have edited any article in common with P_u **do**
$S_{uv} \leftarrow J(u, v)$
if $S_{uv} > Z$ **then**
add v to $retrievedList$ with $v.rsv = u.rsv \cdot S_{uv}$
end if
end for
end for

Annotations can help to identify the correct articles to consider (see Section 2), knowledge taxonomies can help in finding the correct experts, and ontologies can help in disambiguating multi senses topics.

4.1 Using Ontologies as Expertise Taxonomies

The expert finding task is usually performed in Enterprises where the significant knowledge areas are limited. For this reason the expert finding system usually adopt customized taxonomies to model the organization's most important knowledge areas [2].

On the Web the expertise areas covered are much more wide than in an Enterprise. For this reason finding expert on the Web will require much more effort to manually develop a universal expertise taxonomy. We propose to use the Yago ontology [15], a combination of notions from WordNet and Wikipedia, to model the expertise and to identify the knowledge areas used to describe people's knowledge. In this way we can better define the expert profiles according to Yago. For example, knowing that "Macintosh computer" *is a subclass of* "Computers" can help the system when there are no results for the query "Find an expert on Computer". The system can proceed looking for experts in the relative subcategories. More, if we know that "Eclipse" *is a* "Java tool" we can assume that an expert on Eclipse will be an expert (with score proportional to the number of children of the class "Java tool") on Java tools.

4.2 Using WordNet to Disambiguate Expertise Topics

Differently from the Enterprise context where the topic is narrow and clearly defined, in such a wide topics set as the one maintained in Wikipedia there is one more problem to take into account: the topic ambiguity. Multi sense terms might represent topics of expertise. For example, an expert on "Bank" might be expert on only one of the several senses of this noun: slope/incline | financial institution/organization | ridge | array | reserve | ...

Using the algorithm JIGSAW [14] for word sense disambiguation (WSD) we can disambiguate between different topics of expertise. JIGSAW calculates the similarity between each candidate meaning for an ambiguous word and all the meanings in its context defined as words with the same POS tag in the same sentence. The similarity is calculated as inversely proportional to path length between concepts in the WordNet IS-A hierarchy. The assumption in this case is that the appropriate meaning belongs to a similar/same concept as words in the context belong to. For example, if the sentence "John Doe manages the Citizen Bank that has good availability of cash." is an evidence of the expertise on the topic "Bank", we can disambiguate its sense using the context and, in this case, the meaning of "cash". The distance between all the meanings of "Bank" and all the meanings of the nouns in the context (defined as a window of text surrounding the term) can be used in order to find the intended sense. We can then add the sense "financial institution" to the expertise profile of the candidate "John Doe".

It is also possible to use co-occurrence statistics to improve the quality of the profiles. If we take a user profile P_u we can disambiguate the topics looking at the context in the related articles. For example, according to the profile, the user u is an expert on "Jaguar" and we find that in the articles considered in his profile the word "Car" often co-occur with the word "Jaguar". In this way we can add the topic "Car" to the expertises of u always with the final goal of disambiguation.

5 Evaluation Considerations

We propose to build an expert search system which makes use of the knowledge present in Wikipedia. As described in this paper it is possible to implement such a system using several techniques. We now want to discuss about the effectiveness evaluation aspects.

The present ways to evaluate the performances of a retrieval system are based on the standard proposed for the first time by Cleverdon with the Cranfield Experiments in 1966 [5]. Unfortunately, it is not possible to perform such experiments in our scenario given that there is the need for relevance assessments which represent the ground truth for a list of queries. We can not use any of the IR effectiveness evaluation metrics [8] because there are neither relevance judgements available on the Wikipedia collection nor a list of queries to run.

The most relevant initiatives to our task are SemEval [1], but the problem here is that the evaluated task is people name disambiguation and not expert finding; and Inex [9] which uses an XML Wikipedia collection but it does not consider the expert finding task. A possible solution would be to ask to voluntary Wikipedia users, which we will do in the future, about their expertises in order to have a sort of ground truth.

6 Conclusions

In this paper we proposed algorithms to build expert profiles using Wikipedia. We proposed to search for experts both in the content of Wikipedia and among its users. We also proposed some techniques which make use of semantics to improve the results disambiguating and extending the search.

As future steps we aim at build a systems which makes use of these techniques and test it on Wikipedia also asking some (voluntary) Wikipedia users to evaluate the performances of our system.

Acknowledgments. This work was supported by the Nepomuk project funded by the European Commission under the 6th Framework Programme (IST Contract No. 027705). We thank Paul-Alexandru Chirita for his useful comments and advices on our work.

References

1. J. Artiles, J. Gonzalo, and S. Sekine. The SemEval-2007 WePS Evaluation: Establishing a benchmark for the Web People Search Task. *Proceedings of Semeval*, 2007.
2. I. Becerra-Fernandez. Searching for experts on the Web: A review of contemporary expertise locator systems. *ACM Transactions on Internet Technology (TOIT)*, 6(4):333–355, 2006.
3. John G. Breslin, Uldis Bojars, Boanerges Aleman-Meza, Harold Boley, Malgorzata Mochol, Lyndon J. B. Nixon, Axel Polleres, and Anna V. Zhdanova. Finding experts using Internet-based discussions in online communities and associated social networks. *First International ExpertFinder Workshop, Berlin, Germany, January 16, 2007*, 2007.
4. C.S. Campbell, P.P. Maglio, A. Cozzi, and B. Dom. Expertise identification using email communications. *Proceedings of the 12th ACM Conference on Information and Knowledge Management (CIKM'03)*, pages 528–531, 2003.
5. C.W. Cleverdon, J. Mills, and E.M. Keen. Factors determining the performance of indexing systems,(Volume 1: Design). *Cranfield: College of Aeronautics*, 1966.
6. D. Cosley, D. Frankowski, L. Terveen, and J. Riedl. SuggestBot: using intelligent task routing to help people find work in wikipedia. *Proceedings of the 12th international conference on Intelligent user interfaces*, pages 32–41, 2007.
7. N. Craswell, D. Hawking, A. Vercoustre, and P. Wilkins. P@noptic Expert: Searching for Experts not just for Documents. *Ausweb, 2001*.
8. G. Demartini and S. Mizzaro. A classification of IR effectiveness metrics. In *ECIR*, pages 488–491, 2006.
9. N. Fuhr, N. Govert, G. Kazai, and M. Lalmas. INEX: INitiative for the Evaluation of XML retrieval. *Proceedings of the SIGIR 2002 Workshop on XML and Information Retrieval*, 2002.
10. A. Hogan and A. Harth. The ExpertFinder Corpus 2007 for the Benchmarking and Development of Expert-Finding Systems. *First International ExpertFinder Workshop, Berlin, Germany, January 16, 2007*, 2007.
11. J.M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.
12. Jie Li, Harold Boley, Virendrakumar C. Bhavsar, and Jing Mei. Expert finding for eCollaboration using FOAF with RuleML rules. *Montreal Conference on eTechnologies (MCTECH)*, 2006.
13. L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web, 1998.
14. G. Semeraro, M. Degemmis, P. Lops, and P. Basile. Combining learning and word sense disambiguation for intelligent user profiling. *Twentieth International Joint Conference on Artificial Intelligence*, 2007.
15. F.M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. *Proceedings of the 16th international conference on World Wide Web*, pages 697–706, 2007.
16. J. Zhang, M.S. Ackerman, and L. Adamic. Expertise Networks in Online Communities: Structure and Algorithms. *Proceedings of the 16th international conference on World Wide Web*, pages 221–230, 2007.
17. J. Zhu, AL Gonçalves, VS Uren, E. Motta, and R. Pacheco. Mining Web Data for Competency Management. *Web Intelligence '05*, pages 94–100, 2005.